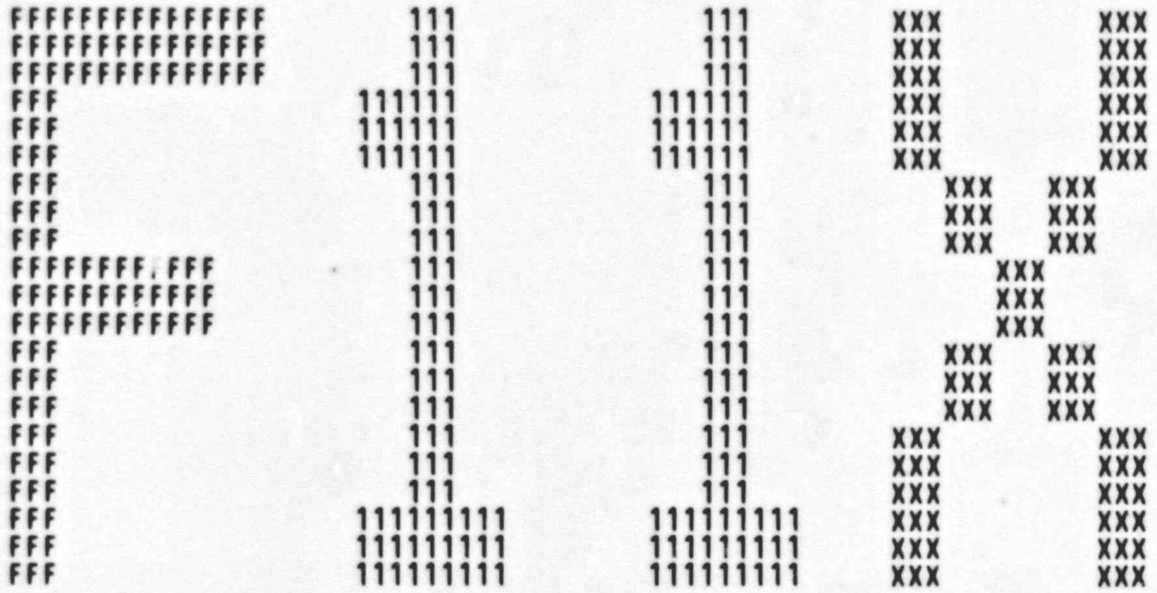

_$2

Symb

IOCS
IO_C
IO_C
IO_D
IO_F
IO_S
KICL

KILL
KILL
LB_B
LB_D
LB_F
LB_K
LB_L
LOCA
LOCA
LOCK

LOCK
LOCK
LOCK
LOC_
LOC_
L_CO
L_CO
L_DA
L_DA
MAIN
MAKE
MAKE
MAKE
MAKE
MAKE
MAKE

MAKE
MAKE
MAP_
MAP_

MAP
MARK
MARK
MARK
MARK
MARK

FILEID**FILUTL

```
FFFFFFFFFF    IIIIII    LL          UU      UU  TTTTTTTTTT  LL
FFFFFFFFFF    IIIIII    LL          UU      UU  TTTTTTTTTT  LL
FF              II      LL          UU      UU      TT      LL
FF              II      LL          UU      UU      TT      LL
FF              II      LL          UU      UU      TT      LL
FF              II      LL          UU      UU      TT      LL
FFFFFFFF        II      LL          UU      UU      TT      LL
FFFFFFFF        II      LL          UU      UU      TT      LL
FF              II      LL          UU      UU      TT      LL
FF              II      LL          UU      UU      TT      LL
FF              II      LL          UU      UU      TT      LL                .....
FF              II      LL          UU      UU      TT      LL                .....
FF            IIIIII    LLLLLLLLLL  UUUUUUUUUU      TT      LLLLLLLLLL        .....
FF            IIIIII    LLLLLLLLLL  UUUUUUUUUU      TT      LLLLLLLLLL        .....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

FIL
V04

```
;     1        0001  0 MODULE FILUTL (
;     2        0002  0                 LANGUAGE (BLISS32),
;     3        0003  0                 IDENT = 'V04-000'
;     4        0004  0                 ) =
;     5        0005  1 BEGIN
;     6        0006  1
;     7        0007  1 !
;     8        0008  1 !**********************************************************************
;     9        0009  1 !*                                                                    *
;    10        0010  1 !*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                           *
;    11        0011  1 !*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.            *
;    12        0012  1 !*  ALL RIGHTS RESERVED.                                             *
;    13        0013  1 !*                                                                    *
;    14        0014  1 !*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
;    15        0015  1 !*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
;    16        0016  1 !*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
;    17        0017  1 !*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
;    18        0018  1 !*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
;    19        0019  1 !*  TRANSFERRED.                                                       *
;    20        0020  1 !*                                                                    *
;    21        0021  1 !*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
;    22        0022  1 !*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
;    23        0023  1 !*  CORPORATION.                                                       *
;    24        0024  1 !*                                                                    *
;    25        0025  1 !*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
;    26        0026  1 !*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.            *
;    27        0027  1 !*                                                                    *
;    28        0028  1 !*                                                                    *
;    29        0029  1 !**********************************************************************
;    30        0030  1
;    31        0031  1 !++
;    32        0032  1 !
;    33        0033  1 ! FACILITY:  F11ACP Structure Level 2
;    34        0034  1 !
;    35        0035  1 ! ABSTRACT:
;    36        0036  1 !
;    37        0037  1 !       This module contains routines used to access random files by the
;    38        0038  1 !       ACP itself.
;    39        0039  1 !
;    40        0040  1 ! ENVIRONMENT:
;    41        0041  1 !
;    42        0042  1 !       STARLET operating system, including privileged system services
;    43        0043  1 !       and internal exec routines.
;    44        0044  1 !
;    45        0045  1 !--
;    46        0046  1 !
;    47        0047  1 !
;    48        0048  1 ! AUTHOR:  Andrew C. Goldstein,  CREATION DATE:  22-May-1978  19:13
;    49        0049  1 !
;    50        0050  1 ! MODIFIED BY:
;    51        0051  1 !
;    52        0052  1 !       V03-015 CDS0010         Christian D. Saether    14-Aug-1984
;    53        0053  1 !               Modify handling of extension fcbs.
;    54        0054  1 !
;    55        0055  1 !       V03-014 CDS0009         Christian D. Saether     6-Aug-1984
;    56        0056  1 !               Correctly deal with serializing on a lock we already had.
;    57        0057  1 !               Add handler for the open_file routine to correctly
```

```
;    58          0058  1 !                clean up after errors in the open_file routine.
;    59          0059  1 !
;    60          0060  1 !        V03-013 LMP0275         L. Mark Pilant,         25-Jul-1984  15:50
;    61          0061  1 !                Don't try to delete an uninitialized ACL.
;    62          0062  1 !
;    63          0063  1 !        V03-012 CDS0008         Christian D. Saether    19-Apr-1984
;    64          0064  1 !                Use REFCNT instead of ACNT.
;    65          0065  1 !                Modify access arbitration.
;    66          0066  1 !
;    67          0067  1 !        V03-011 ACG0415         Andrew C. Goldstein,    5-Apr-1984  21:33
;    68          0068  1 !                Interface change to ACL_DELETEACL
;    69          0069  1 !
;    70          0070  1 !        V03-010 ACG0408         Andrew C. Goldstein,    20-Mar-1984  17:47
;    71          0071  1 !                Make APPLY_RVN and DEFAULT_RVN macros
;    72          0072  1 !
;    73          0073  1 !        V03-009 CDS0007         Christian D. Saether    23-Feb-1984
;    74          0074  1 !                Eliminate use of FLUSH_LOCK_BASIS.
;    75          0075  1 !                Replace with TOSS_CACHE_DATA.
;    76          0076  1 !
;    77          0077  1 !        V03-008 CDS0006         Christian D. Saether    18-Jan-1984
;    78          0078  1 !                Modify interface to APPLY_RVN.
;    79          0079  1 !
;    80          0080  1 !        V03-007 CDS0005         Christian D. Saether    30-Dec-1983
;    81          0081  1 !                Use L_NORM linkage and BIND_COMMON macro.
;    82          0082  1 !
;    83          0083  1 !        V03-006 CDS0004         Christian D. Saether     7-Dec-1983
;    84          0084  1 !                Remove call to REMOVE_FCB and do the REMQUE inline.
;    85          0085  1 !
;    86          0086  1 !        V03-005 CDS0003         Christian D. Saether    14-Sep-1983
;    87          0087  1 !                Modify SERIAL_FILE interface.  Use RELEASE_SERIAL_LOCK
;    88          0088  1 !                routine to dequeue serialization lock.
;    89          0089  1 !
;    90          0090  1 !        V03-004 CDS0002         Christian D. Saether    19-Jun-1983
;    91          0091  1 !                Until further work is done with buffer caching,
;    92          0092  1 !                flush all buffers from the cache when closing internal file.
;    93          0093  1 !                This fixes a bug where getting location information for
;    94          0094  1 !                VBN placement leaves a header in the cache and the file
;    95          0095  1 !                serialization lock is released.
;    96          0096  1 !
;    97          0097  1 !        V03-003 CDS0001         Christian D. Saether     5-May-1983
;    98          0098  1 !                Add xqp synchronization of file processing (SERIAL_FILE)
;    99          0099  1 !                and xqp access arbitration (ACCESS_LOCK) calls.
;   100          0100  1 !
;   101          0101  1 !        V03-02  LMP0059         L. Mark Pilant,         7-Jan-1983  12:05
;   102          0102  1 !                Always create and link in an FCB when accessing a file.  This
;   103          0103  1 !                eliminates a lot of special case handling.
;   104          0104  1 !
;   105          0105  1 !        V03-001 LMP0037         L. Mark Pilant,         28-Jun-1982  15:10
;   106          0106  1 !                Remove the addressing mode module switch.
;   107          0107  1 !
;   108          0108  1 !        V02-006 ACG0259         Andrew C. Goldstein,    27-Jan-1982  20:38
;   109          0109  1 !                Change to longword external addressing
;   110          0110  1 !
;   111          0111  1 !        V02-004 LMP0003         L. Mark Pilant,         8-Dec-1981  11:31
;   112          0112  1 !                Make sure the primary window was actually created.  It may
;   113          0113  1 !                not have been due to the byte limit quota being exceeded.
;   114          0114  1 !
```

```
;   115     0115  1 !       B0104   ACG0112         Andrew C. Goldstein,    15-Jan-1980  22:55
;   116     0116  1 !               Limit data read to file's EOF
;   117     0117  1 !
;   118     0118  1 !       B0103   ACG0092         Andrew C. Goldstein,    6-Dec-1979  19:23
;   119     0119  1 !               Set proper RVN on file being opened
;   120     0120  1 !
;   121     0121  1 !       B0102   ACG0008         Andrew C. Goldstein,  18-Dec-1978  22:57
;   122     0122  1 !       Add map only access for placement use, support multi-header files
;   123     0123  1 !
;   124     0124  1 !       B0101   ACG0003         Andrew C. Goldstein,  10-Nov-1978  19:01
;   125     0125  1 !       Add multi-volume support, restrict to single header files
;   126     0126  1 !
;   127     0127  1 !       B0100   ACG00001        Andrew C. Goldstein,  10-Oct-1978  20:00
;   128     0128  1 !       Previous revision history moved to [F11B.SRC]F11B.REV
;   129     0129  1 !**
;   130     0130  1
;   131     0131  1
;   132     0132  1 LIBRARY 'SYS$LIBRARY:LIB.L32';
;   133     0133  1 REQUIRE 'SRC$:FCPDEF.B32';
;   134     1124  1
;   135     1125  1
;   136     1126  1 FORWARD ROUTINE
;   137     1127  1         OPEN_FILE        : L_NORM,        ! open a file
;   138     1128  1         OPEN_FILE_HANDLER : L_NORM,       ! error handling for open_file
;   139     1129  1         READ_DATA        : L_NORM,        ! read data from file
;   140     1130  1         CLOSE_FILE       : L_NORM NOVALUE; ! close a file
```

```
;  142          1131  1 GLOBAL ROUTINE OPEN_FILE (FID, WRITE) : L_NORM =
;  143          1132  1
;  144          1133  1 !++
;  145          1134  1 !
;  146          1135  1 ! FUNCTIONAL DESCRIPTION:
;  147          1136  1 !
;  148          1137  1 !       This routine opens the file of the given file ID. It constructs an
;  149          1138  1 !       FCB and window and returns the address of the latter.
;  150          1139  1 !
;  151          1140  1 !
;  152          1141  1 ! CALLING SEQUENCE:
;  153          1142  1 !       OPEN_FILE (ARG1, ARG2)
;  154          1143  1 !
;  155          1144  1 ! INPUT PARAMETERS:
;  156          1145  1 !       ARG1: address of file ID of file to open
;  157          1146  1 !       ARG2: = 0 to open read only
;  158          1147  1 !               1 to open read/write
;  159          1148  1 !               2 to bypass interlocks (just map the file)
;  160          1149  1 !
;  161          1150  1 ! IMPLICIT INPUTS:
;  162          1151  1 !       NONE
;  163          1152  1 !
;  164          1153  1 ! OUTPUT PARAMETERS:
;  165          1154  1 !       NONE
;  166          1155  1 !
;  167          1156  1 ! IMPLICIT OUTPUTS:
;  168          1157  1 !       PRIMARY_FCB: address of FCB created or found
;  169          1158  1 !       CURRENT_WINDOW: address of window created
;  170          1159  1 !
;  171          1160  1 ! ROUTINE VALUE:
;  172          1161  1 !       address of window created
;  173          1162  1 !
;  174          1163  1 ! SIDE EFFECTS:
;  175          1164  1 !       FCB and window created
;  176          1165  1 !
;  177          1166  1 !--
;  178          1167  1
;  179          1168  2 BEGIN
;  180          1169  2
;  181          1170  2 MAP
;  182          1171  2         FID             : REF BBLOCK;   ! file ID arg
;  183          1172  2
;  184          1173  2 LOCAL
;  185          1174  2         FCB_CREATED,                    ! flag indicating FCB creation
;  186          1175  2         FCB             : REF BBLOCK,   ! file control block address
;  187          1176  2         WINDOW          : REF BBLOCK,   ! window address
;  188          1177  2         HEADER          : REF BBLOCK;   ! file header address
;  189          1178  2
;  190          1179  2 BIND_COMMON;
;  191          1180  2
;  192          1181  2 EXTERNAL ROUTINE
;  193          1182  2         REBLD_PRIM_FCB  : L_NORM NOVALUE, ! rebuild primary fcb from header
;  194          1183  2         BUILD_EXT_FCBS  : L_NORM NOVALUE, ! build extension fcbs
;  195          1184  2         ARBITRATE_ACCESS : [ JSB_2ARGS, ! arbitrate file access
;  196          1185  2         CONV_ACCLOCK    : L_NORM,       ! convert file access lock
;  197          1186  2         SERIAL_FILE     : L_NORM,       ! file processing interlock
;  198          1187  2         SWITCH_VOLUME   : L_NORM,       ! switch to correct volume
```

```
;  199          1188  2          SEARCH_FCB      : L_NORM,       ! search for FCB of file
;  200          1189  2          READ_HEADER     : L_NORM,       ! read file header
;  201          1190  2          CREATE_FCB      : L_NORM,       ! create a file control block
;  202          1191  2          CREATE_WINDOW   : L_NORM;       ! create a file window
;  203          1192  2
;  204          1193  2  ENABLE OPEN_FILE_HANDLER;
;  205          1194  2
;  206          1195  2  ! The current uses of this routine (as of 3b) are
;  207          1196  2  ! 1) BADSCN calls it to get r/w access to the badlog file
;  208          1197  2  ! 2) GET_LOC calls it with bypass to get mapping info for related file placement
;  209          1198  2  ! 3) CREATE calls it with bypass to get previous version attributes for
;  210          1199  2  ! propagation
;  211          1200  2  !
;  212          1201  2  ! There is a small possibility of deadlock on the placement use because of
;  213          1202  2  ! the file serialization lock.  If two processes simultaneously do placed
;  214          1203  2  ! allocation on two separate files, and each specifies the other as the
;  215          1204  2  ! file to be placed near, one could deadlock.
;  216          1205  2  !
;  217          1206  2
;  218          1207  2  ! Initialize impure cells that drive the cleanup in the handler.
;  219          1208  2  !
;  220          1209  2
;  221          1210  2  STSFLGS [STS_HAD_LOCK] = 0;
;  222          1211  2  STSFLGS [STS_KEEP_LOCK] = 0;
;  223          1212  2  PRIMARY_FCB = 0;
;  224          1213  2  PRIM_LCKINDX = 0;
;  225          1214  2
;  226          1215  2  ! Switch context to the volume of the specified RVN.
;  227          1216  2  !
;  228          1217  2
;  229          1218  2  APPLY_RVN (FID[FID$W_RVN], .CURRENT_RVN);
;  230          1219  2  SWITCH_VOLUME (.FID[FID$W_RVN]);
;  231          1220  2
;  232          1221  2  ! Interlock processing on this file.
;  233          1222  2  ! There is an assumption made in the way that this lock is handled
;  234          1223  2  ! that no other serial_file calls will be made before a close_file
;  235          1224  2  ! is done on this file.  That is because the sts_had_lock flag will
;  236          1225  2  ! be set by serial_file and we are going to use that flag to determine
;  237          1226  2  ! whether to release this lock in close_file.
;  238          1227  2  !
;  239          1228  2
;  240          1229  2  PRIM_LCKINDX = SERIAL_FILE (.FID);
;  241          1230  2
;  242          1231  2  IF .STSFLGS [STS_HAD_LOCK]
;  243          1232  2  THEN
;  244          1233  2      STSFLGS [STS_KEEP_LOCK] = 1;
;  245          1234  2
;  246          1235  2  ! Search the FCB list for the given file ID. If found, arbitrate access
;  247          1236  2  ! interlocks. Note that if we create an FCB, we do not bother with access
;  248          1237  2  ! counts, etc., since it will disappear at the end of this call.
;  249          1238  2  !
;  250          1239  2
;  251          1240  2  FCB = SEARCH_FCB (.FID);
;  252          1241  2
;  253          1242  2  HEADER = READ_HEADER (.FID, .FCB);
;  254          1243  2  FCB_CREATED = 0;
;  255          1244  2  IF .FCB EQL 0
```

```
;  256     1245  2 THEN
;  257     1246  2     BEGIN
;  258     1247  2     FCB_CREATED = 1;
;  259     1248  2     FCB = KERNEL_CALL (CREATE_FCB, .HEADER);
;  260     1249  2     END;
;  261     1250  2
;  262     1251  2 PRIMARY_FCB = .FCB;
;  263     1252  2
;  264     1253  2 IF .WRITE NEQ 2
;  265     1254  2 THEN
;  266     1255  3     BEGIN
;  267     1256  3     LOCAL
;  268     1257  3         CURR_LKMODE;
;  269     1258  3
;  270     1259  3     CURR_LKMODE = .FCB [FCB$B_ACCLKMODE];
;  271     1260  3
;  272     1261  3     IF NOT ARBITRATE_ACCESS (IF .WRITE THEN FIB$M_WRITE ELSE 0, .FCB)
;  273     1262  3     THEN ERR_EXIT (SS$_ACCONFLICT);
;  274     1263  3
;  275     1264  3     CONV_ACCLOCK (.CURR_LKMODE, .FCB);
;  276     1265  2     END;
;  277     1266  2
;  278     1267  2 ! By setting this cleanup flag, further error recovery is done in
;  279     1268  2 ! the error_cleanup routine, not by the open_file_handler.
;  280     1269  2 !
;  281     1270  2
;  282     1271  2 CLEANUP_FLAGS[CLF_CLOSEFILE] = 1;
;  283     1272  2
;  284     1273  2 CURRENT_WINDOW = WINDOW = CREATE_WINDOW (0, 0, .HEADER, 0, .FCB);
;  285     1274  2
;  286     1275  2 IF .CURRENT_WINDOW EQL 0 THEN ERR_EXIT (SS$_EXBYTLM);
;  287     1276  2
;  288     1277  2 ! If the file is multi-header, read the extension headers and create
;  289     1278  2 ! extension FCB's as necessary. Finally read back the primary header.
;  290     1279  2 !
;  291     1280  2
;  292     1281  2 IF .FCB_CREATED
;  293     1282  2 THEN
;  294     1283  2     BUILD_EXT_FCBS (.HEADER)
;  295     1284  2 ELSE
;  296     1285  2     IF .FCB [FCB$V_STALE]
;  297     1286  2     THEN
;  298     1287  3         BEGIN
;  299     1288  3
;  300     1289  3         REBLD_PRIM_FCB (.FCB, .HEADER);
;  301     1290  3
;  302     1291  3         BUILD_EXT_FCBS (.HEADER);
;  303     1292  3
;  304     1293  2         END;
;  305     1294  2
;  306     1295  2 RETURN .WINDOW;
;  307     1296  2
;  308     1297  1 END;                                    ! end of routine OPEN_FILE

                                                        .TITLE  FILUTL
                                                        .IDENT  \V04-000\
```

```
                                        .EXTRN  REBLD_PRIM_FCB, BUILD_EXT_FCBS
                                        .EXTRN  ARBITRATE_ACCESS
                                        .EXTRN  CONV_ACCLOCK, SERIAL_FILE
                                        .EXTRN  SWITCH_VOLUME, SEARCH_FCB
                                        .EXTRN  READ_HEADER, CREATE_FCB
                                        .EXTRN  CREATE_WINDOW

                                        .PSECT  $CODE$,NOWRT,2

                          003C 00000    .ENTRY  OPEN_FILE, Save R2,R3,R4,R5                     ; 1131
                6D      00E2 CF  DE 00002     MOVAL   12$, (FP)                               ; 1177
            A6  AA           06  8A 00007     BICB2   #6, -90(BASE)                           ; 1211
                         08  AA  D4 0000B     CLRL    8(BASE)                                 ; 1212
                         18  AA  D4 0000E     CLRL    24(BASE)                                ; 1213
                50           04  AC  D0 00011     MOVL    FID, R0                             ; 1218
                             04  A0  95 00015     TSTB    4(R0)                               ;
                             05  12  00018     BNEQ    1$                                  ;
                04  A0           A0  AA  90 0001A     MOVB    -96(BASE), 4(R0)                ;
                50           04  AC  D0 0001F 1$:     MOVL    FID, R0                         ;
                01           04  A0  91 00023     CMPB    4(R0), #1                       ;
                             08  12  00027     BNEQ    2$                                  ;
                         A0  AA  D5 00029     TSTL    -96(BASE)                           ;
                             03  12  0002C     BNEQ    2$                                  ;
                             04  A0  94 0002E     CLRB    4(R0)                           ;
                50           04  AC  D0 00031 2$:     MOVL    FID, R0                         ; 1219
                7E           04  A0  3C 00035     MOVZWL  4(R0), -(SP)                    ;
          0000G CF               01  FB 00039     CALLS   #1, SWITCH_VOLUME               ;
                             04  AC  DD 0003E     PUSHL   FID                             ; 1229
          0000G CF               01  FB 00041     CALLS   #1, SERIAL_FILE                 ;
            18  AA               50  D0 00046     MOVL    R0, 24(BASE)                    ;
    04      A6  AA               01  E1 0004A     BBC     #1, -90(BASE), 3$               ; 1231
            A6  AA               04  88 0004F     BISB2   #4, -90(BASE)                   ; 1233
                         04  AC  DD 00053 3$:     PUSHL   FID                             ; 1240
          0000G CF               01  FB 00056     CALLS   #1, SEARCH_FCB                  ;
                52               50  D0 0005B     MOVL    R0, FCB                         ;
                                 52  DD 0005E     PUSHL   FCB                             ; 1242
                             04  AC  DD 00060     PUSHL   FID                             ;
          0000G CF               02  FB 00063     CALLS   #2, READ_HEADER                 ;
                55               50  D0 00068     MOVL    R0, HEADER                      ;
                                 54  D4 0006B     CLRL    FCB_CREATED                     ; 1243
                                 52  D5 0006D     TSTL    FCB                             ; 1244
                                 0D  12 0006F     BNEQ    4$                              ;
                54               01  D0 00071     MOVL    #1, FCB_CREATED                 ; 1247
                                 55  DD 00074     PUSHL   HEADER                          ; 1248
          0000G CF               01  FB 00076     CALLS   #1, CREATE_FCB                  ;
                52               50  D0 0007B     MOVL    R0, FCB                         ;
            08  AA               52  D0 0007E 4$:     MOVL    FCB, 8(BASE)                ; 1251
                02           08  AC  D1 00082     CMPL    WRITE, #2                       ; 1253
                             28  13  00086     BEQL    8$                              ;
                53           0B  A2  9A 00088     MOVZBL  11(FCB), CURR_LKMODE            ; 1259
                07           08  AC  E9 0008C     BLBC    WRITE, 5$                       ; 1261
                50       0100 8F  3C 00090     MOVZWL  #256, R0                        ;
                                 02  11 00095     BRB     6$                              ;
                                 50  D4 00097 5$:     CLRL    R0                          ;
                51               52  D0 00099 6$:     MOVL    FCB, R1                     ;
                             0000G 30  0009C     BSBW    ARBITRATE_ACCESS                ;
```

```
                      05              50  E8  0009F          BLBS    R0, 7$                                          ;
                            0800      8F  BF  000A2          CHMU    #2048                                           ; 1262
                                          04  000A6          RET                                                     ;
                                      52  DD  000A7  7$:     PUSHL   FCB                                             ; 1264
                                      53  DD  000A9          PUSHL   CURR_LKMODE                                     ;
                0000G                 CF  02  FB  000AB      CALLS   #2, CONV_ACCLOCK                                ;
                  03                  AA  01  88  000B0  8$: BISB2   #1, 3(BASE)                                     ; 1271
                                      52  DD  000B4          PUSHL   FCB                                             ; 1273
                                      7E  D4  000B6          CLRL    -(SP)                                           ;
                                      55  DD  000B8          PUSHL   HEADER                                          ;
                                      7E  7C  000BA          CLRQ    -(SP)                                           ;
                0000G                 CF  05  FB  000BC      CALLS   #5, CREATE_WINDOW                               ;
                                      53  50  D0  000C1      MOVL    R0, WINDOW                                      ;
                  0C                  AA  53  D0  000C4      MOVL    WINDOW, 12(BASE)                                ;
                                      05  12  000C8          BNEQ    9$                                              ; 1275
                            2A14      8F  BF  000CA          CHMU    #10772                                          ;
                                          04  000CE          RET                                                     ;
                      0B                  54  E8  000CF  9$: BLBS    FCB_CREATED, 10$                                ; 1281
                      0E            23    A2  E9  000D2      BLBC    35(FCB), 11$                                    ; 1285
                                          24  BB  000D6      PUSHR   #^M<R2,R5>                                      ; 1289
                0000G                 CF  02  FB  000D8      CALLS   #2, REBLD_PRIM_FCB                              ;
                                      55  DD  000DD  10$:    PUSHL   HEADER                                          ; 1291
                0000G                 CF  01  FB  000DF      CALLS   #1, BUILD_EXT_FCBS                              ;
                      50                  53  D0  000E4  11$: MOVL    WINDOW, R0                                      ; 1295
                                          04  000E7          RET                                                     ; 1297
                                    0000      000E8  12$:     .WORD   Save nothing                                    ; 1177
                                      7E  D4  000EA          CLRL    -(SP)                                           ;
                                      5E  DD  000EC          PUSHL   SP                                              ;
                      7E            04  AC  7D  000EE      MOVQ    4(AP), -(SP)                                    ;
                0000V                 CF  03  FB  000F2      CALLS   #3, OPEN_FILE_HANDLER                           ;
                                          04  000F7          RET                                                     ;

; Routine Size:  248 bytes,    Routine Base:  $CODE$ + 0000
```

```
;  310     1298  1 ROUTINE OPEN_FILE_HANDLER (SIGNAL, MECHANISM) : L_NORM =
;  311     1299  1
;  312     1300  1 !++
;  313     1301  1 !
;  314     1302  1 ! FUNCTIONAL DESCRIPTION:
;  315     1303  1 !
;  316     1304  1 !       Clean up from aborted open_file.  Specifically, get rid of
;  317     1305  1 !       the fcb and serialization lock if we did not previously
;  318     1306  1 !       hold the serialization lock.
;  319     1307  1 !
;  320     1308  1 !--
;  321     1309  1
;  322     1310  2 BEGIN
;  323     1311  2
;  324     1312  2 MAP
;  325     1313  2         SIGNAL  : REF BBLOCK;
;  326     1314  2
;  327     1315  2 BIND_COMMON;
;  328     1316  2
;  329     1317  2 EXTERNAL ROUTINE
;  330     1318  2         NUKE_HEAD_FCB   : L_NORM NOVALUE, ! cleanup and deallocate prim fcb
;  331     1319  2         RELEASE_SERIAL_LOCK : L_NORM NOVALUE,
;  332     1320  2         SET_DIRINDX     : L_JSB_1ARG;
;  333     1321  2
;  334     1322  2 IF .SIGNAL [CHF$L_SIG_NAME] NEQ SS$_CMODUSER
;  335     1323  2     OR .CLEANUP_FLAGS [CLF_CLOSEFILE]
;  336     1324  2     OR .PRIM_LCKINDX EQL 0
;  337     1325  2     OR .STSFLGS [STS_KEEP_LOCK]
;  338     1326  2 THEN
;  339     1327  2     RETURN SS$_RESIGNAL;
;  340     1328  2
;  341     1329  2 IF .PRIMARY_FCB NEQ 0
;  342     1330  2 THEN
;  343     1331  2     IF .PRIMARY_FCB [FCB$W_REFCNT] EQL 0
;  344     1332  2     THEN
;  345     1333  2         IF NOT SET_DIRINDX (.PRIMARY_FCB)
;  346     1334  2         THEN
;  347     1335  2             NUKE_HEAD_FCB (.PRIMARY_FCB);
;  348     1336  2
;  349     1337  2 PRIMARY_FCB = 0;
;  350     1338  2
;  351     1339  2 IF .PRIM_LCKINDX NEQ 0
;  352     1340  2 THEN
;  353     1341  2     RELEASE_SERIAL_LOCK (.PRIM_LCKINDX);
;  354     1342  2
;  355     1343  2 PRIM_LCKINDX = 0;
;  356     1344  2
;  357     1345  2 SS$_RESIGNAL
;  358     1346  2
;  359     1347  1 END;                    ! of routine OPEN_FILE_HANDLER


                                                                .EXTRN  NUKE_HEAD_FCB, RELEASE_SERIAL_LOCK
                                                                .EXTRN  SET_DIRINDX

                                        000C 00000 OPEN_FILE_HANDLER:
```

```
                                50      04    AC  D0 00002          .WORD   Save R2,R3                              ; 1298
                                                                     MOVL    SIGNAL, R0                              ; 1322
                          00000424  8F      04  A0  D1 00006          CMPL    4(R0), #1060                            ;
                                                3A  12 0000E          BNEQ    3$                                      ;
                                36      03    AA  E8 00010          BLBS    3(BASE), 3$                             ; 1323
                                        18    AA  D5 00014          TSTL    24(BASE)                                ; 1324
                                                31  13 00017          BEQL    3$                                      ;
                      2C          A6    AA        02  E0 00019          BBS     #2, -90(BASE), 3$                       ; 1325
                                50      08    AA  D0 0001E          MOVL    8(BASE), R0                             ; 1329
                                                13  13 00022          BEQL    1$                                      ;
                                        18    A0  B5 00024          TSTW    24(R0)                                  ; 1331
                                                0E  12 00027          BNEQ    1$                                      ;
                                            0000G 30 00029          BSBW    SET_DIRINDX                             ; 1333
                                08              50  E8 0002C          BLBS    R0, 1$                                  ;
                                        08    AA  DD 0002F          PUSHL   8(BASE)                                 ; 1335
                              0000G     CF        01  FB 00032          CALLS   #1, NUKE_HEAD_FCB                       ;
                                        08    AA  D4 00037 1$:      CLRL    8(BASE)                                 ; 1337
                                        18    AA  D5 0003A          TSTL    24(BASE)                                ; 1339
                                                08  13 0003D          BEQL    2$                                      ;
                                        18    AA  DD 0003F          PUSHL   24(BASE)                                ; 1341
                              0000G     CF        01  FB 00042          CALLS   #1, RELEASE_SERIAL_LOCK                 ;
                                        18    AA  D4 00047 2$:      CLRL    24(BASE)                                ; 1343
                                50    0918  8F  3C  0004A 3$:      MOVZWL  #2328, R0                               ; 1347
                                                04  0004F          RET                                             ;

; Routine Size:  80 bytes,    Routine Base:  $CODE$ + 00F8
```

```
;  361     1348  1 GLOBAL ROUTINE READ_DATA (WINDOW, VBN, COUNT) : L_NORM =
;  362     1349  1
;  363     1350  1 !++
;  364     1351  1 !
;  365     1352  1 ! FUNCTIONAL DESCRIPTION:
;  366     1353  1 !
;  367     1354  1 !       This routine reads the specified data block(s) from the file indicated
;  368     1355  1 !       by the given window address. Note that the actual number of blocks
;  369     1356  1 !       read may be less than the number desired due to mapping fragmentation
;  370     1357  1 !       or cache limitations.
;  371     1358  1 !
;  372     1359  1 !
;  373     1360  1 ! CALLING SEQUENCE:
;  374     1361  1 !       READ_DATA (ARG1, ARG2, ARG3)
;  375     1362  1 !
;  376     1363  1 ! INPUT PARAMETERS:
;  377     1364  1 !       ARG1: window address
;  378     1365  1 !       ARG2: starting VBN to read
;  379     1366  1 !       ARG3: count of blocks to read
;  380     1367  1 !
;  381     1368  1 ! IMPLICIT INPUTS:
;  382     1369  1 !       NONE
;  383     1370  1 !
;  384     1371  1 ! OUTPUT PARAMETERS:
;  385     1372  1 !       NONE
;  386     1373  1 !
;  387     1374  1 ! IMPLICIT OUTPUTS:
;  388     1375  1 !       NONE
;  389     1376  1 !
;  390     1377  1 ! ROUTINE VALUE:
;  391     1378  1 !       address of buffer read
;  392     1379  1 !
;  393     1380  1 ! SIDE EFFECTS:
;  394     1381  1 !       block read, window may be turned
;  395     1382  1 !
;  396     1383  1 !--
;  397     1384  1
;  398     1385  2 BEGIN
;  399     1386  2
;  400     1387  2 MAP
;  401     1388  2         WINDOW          : REF BBLOCK;   ! window argument
;  402     1389  2
;  403     1390  2 LOCAL
;  404     1391  2         FCB             : REF BBLOCK,   ! address of file's FCB
;  405     1392  2         LBN,                            ! LBN of starting virtual block
;  406     1393  2         UNMAPPED,                       ! number of desired blocks not mapped
;  407     1394  2         BUFFER          : REF BBLOCK;   ! address of block read
;  408     1395  2
;  409     1396  2 BASE_REGISTER;
;  410     1397  2
;  411     1398  2 EXTERNAL ROUTINE
;  412     1399  2         MAP_VBN         : L_NORM,       ! map virtual to logical
;  413     1400  2         READ_BLOCK      : L_NORM;       ! read a disk block
;  414     1401  2
;  415     1402  2
;  416     1403  2 ! Map the VBN to LBN using the supplied window. If the map fails, return a
;  417     1404  2 ! zero buffer address.
```

```
;  418          1405  2 !
;  419          1406  2
;  420          1407  2 FCB = .WINDOW[WCB$L_FCB];
;  421          1408  2 IF .VBN GTRU .FCB[FCB$L_EFBLK]
;  422          1409  2 THEN RETURN 0;
;  423          1410  2
;  424          1411  2 LBN = MAP_VBN (.VBN, .WINDOW, .COUNT, UNMAPPED);
;  425          1412  2 IF .LBN EQL -1 THEN RETURN 0;
;  426          1413  2
;  427          1414  2 BUFFER = READ_BLOCK (.LBN, .COUNT - .UNMAPPED, DATA_TYPE);
;  428          1415  2 RETURN .BUFFER;
;  429          1416  2
;  430          1417  1 END;                                    ! end of routine READ_DATA


                                                .EXTRN  MAP_VBN, READ_BLOCK

                                0000 00000      .ENTRY  READ_DATA, Save nothing                 ; 1348
                             5E 04   C2 00002   SUBL2   #4, SP                                  ;
                             51 04   AC D0 00005   MOVL    WINDOW, R1                           ; 1407
                             50 18   A1 D0 00009   MOVL    24(R1), FCB                          ;
                          3C 50 08   AC D1 0000D   CMPL    VBN, 60(FCB)                         ; 1408
                                28   1A 00012   BGTRU   1$                                    ;
                                5E   DD 00014   PUSHL   SP                                    ; 1411
                                0C   AC DD 00016   PUSHL   COUNT                              ;
                                51   DD 00019   PUSHL   R1                                   ;
                                08   AC DD 0001B   PUSHL   VBN                                ;
                           0000G CF          04   FB 0001E   CALLS   #4, MAP_VBN              ;
                        FFFFFFFF 8F          50   D1 00023   CMPL    LBN, #-1                 ; 1412
                                             10   13 0002A   BEQL    1$                       ;
                                             04   DD 0002C   PUSHL   #4                       ; 1414
                       7E       0C   AC 04   AE   C3 0002E   SUBL3   UNMAPPED, COUNT, -(SP)  ;
                                             50   DD 00034   PUSHL   LBN                      ;
                           0000G CF          03   FB 00036   CALLS   #3, READ_BLOCK           ;
                                             04   0003B      RET                              ; 1415
                                             50   D4 0003C 1$:    CLRL    R0                 ; 1417
                                             04   0003E      RET                              ;

; Routine Size:  63 bytes,    Routine Base:  $CODE$ + 0148
```

```
;  432     1418  1 GLOBAL ROUTINE CLOSE_FILE (WINDOW) : L_NORM NOVALUE =
;  433     1419  1
;  434     1420  1 !++
;  435     1421  1 !
;  436     1422  1 ! FUNCTIONAL DESCRIPTION:
;  437     1423  1 !
;  438     1424  1 !       This routine closes the file indicated by the supplied window
;  439     1425  1 !       by releasing the window and FCB.
;  440     1426  1 !
;  441     1427  1 !
;  442     1428  1 ! CALLING SEQUENCE:
;  443     1429  1 !       CLOSE_FILE (ARG1)
;  444     1430  1 !
;  445     1431  1 ! INPUT PARAMETERS:
;  446     1432  1 !       ARG1: address of window
;  447     1433  1 !
;  448     1434  1 ! IMPLICIT INPUTS:
;  449     1435  1 !       NONE
;  450     1436  1 !
;  451     1437  1 ! OUTPUT PARAMETERS:
;  452     1438  1 !       NONE
;  453     1439  1 !
;  454     1440  1 ! IMPLICIT OUTPUTS:
;  455     1441  1 !       PRIMARY_FCB: 0
;  456     1442  1 !       CURRENT_WINDOW: 0
;  457     1443  1 !
;  458     1444  1 ! ROUTINE VALUE:
;  459     1445  1 !       NONE
;  460     1446  1 !
;  461     1447  1 ! SIDE EFFECTS:
;  462     1448  1 !       FCB and window deallocated
;  463     1449  1 !
;  464     1450  1 !--
;  465     1451  1
;  466     1452  2 BEGIN
;  467     1453  2
;  468     1454  2 MAP
;  469     1455  2         WINDOW          : REF BBLOCK;   ! window argument
;  470     1456  2
;  471     1457  2 LOCAL
;  472     1458  2         FCB             : REF BBLOCK,   ! FCB of file
;  473     1459  2         WINDOW_SEGMENT  : REF BBLOCK,   ! Address of current window segment
;  474     1460  2         NEXT_SEGMENT    : REF BBLOCK;   ! Address of next window segment
;  475     1461  2
;  476     1462  2 BIND_COMMON;
;  477     1463  2
;  478     1464  2 EXTERNAL ROUTINE
;  479     1465  2         TOSS_CACHE_DATA : L_NORM NOVALUE,
;  480     1466  2         RELEASE_SERIAL_LOCK : L_NORM NOVALUE,
;  481     1467  2         DEALLOCATE      : L_NORM,       ! deallocate back to pool
;  482     1468  2         DEL_EXTFCB      : L_NORM,       ! delete extension FCB's
;  483     1469  2         SET_DIRINDX     : L_JSB_1ARG,   ! test and set for directory fcb
;  484     1470  2         NUKE_HEAD_FCB   : L_NORM NOVALUE; ! cleanup a primary fcb
;  485     1471  2
;  486     1472  2
;  487     1473  2 ! Find the FCB. Deallocate the window, and the FCB if it is not otherwise
;  488     1474  2 ! accessed. Also flush data blocks of the file from the buffer pool.
```

```
;  489          1475  2 !
;  490          1476  2
;  491          1477  2 FCB = .WINDOW[WCB$L_FCB];
;  492          1478  2 TOSS_CACHE_DATA (.PRIM_LCKINDX);
;  493          1479  2
;  494          1480  2 PRIMARY_FCB = 0;
;  495          1481  2 CURRENT_WINDOW = 0;
;  496          1482  2 CLEANUP_FLAGS[CLF_CLOSEFILE] = 0;
;  497          1483  2
;  498          1484  2 WINDOW_SEGMENT = .WINDOW;
;  499          1485  2 DO
;  500          1486  3     BEGIN
;  501          1487  3     NEXT_SEGMENT = .WINDOW_SEGMENT[WCB$L_LINK];
;  502          1488  3     KERNEL_CALL (DEALLOCATE, .WINDOW_SEGMENT);
;  503          1489  3     WINDOW_SEGMENT = .NEXT_SEGMENT;
;  504          1490  3     END
;  505          1491  2 UNTIL .WINDOW_SEGMENT EQL 0;
;  506          1492  2
;  507          1493  2 ! If we already held the serialization lock on this file, we must
;  508          1494  2 ! have it in primary context.  In that case, we will also have
;  509          1495  2 ! remembered the fcb address in primary context, so let's deal with
;  510          1496  2 ! it there.
;  511          1497  2 !
;  512          1498  2
;  513          1499  2 IF .STSFLGS [STS_KEEP_LOCK]
;  514          1500  2 THEN
;  515          1501  3     BEGIN
;  516          1502  3     PRIM_LCKINDX = 0;
;  517          1503  3     RETURN;
;  518          1504  2     END;
;  519          1505  2
;  520          1506  2 IF .FCB[FCB$W_REFCNT] EQL 0
;  521          1507  2 THEN
;  522          1508  2     IF NOT SET_DIRINDX (.FCB)
;  523          1509  2     THEN
;  524          1510  3         BEGIN
;  525          1511  3         DEL_EXTFCB (.FCB);
;  526          1512  3         NUKE_HEAD_FCB (.FCB);
;  527          1513  2         END;
;  528          1514  2
;  529          1515  2 RELEASE_SERIAL_LOCK (.PRIM_LCKINDX);
;  530          1516  2 PRIM_LCKINDX = 0;
;  531          1517  2
;  532          1518  1 END;                                    ! end of routine CLOSE_FILE


                                                        .EXTRN  TOSS_CACHE_DATA
                                                        .EXTRN  DEALLOCATE, DEL_EXTFCB

                                          001C 00000    .ENTRY  CLOSE_FILE, Save R2,R3,R4                       ; 1418
                    50        04    AC    D0 00002    MOVL    WINDOW, R0                                      ; 1477
                    53        18    A0    D0 00006    MOVL    24(R0), FCB                                     ;
                              18    AA    DD 0000A    PUSHL   24(BASE)                                        ; 1478
                0000G CF              01    FB 0000D    CALLS   #1, TOSS_CACHE_DATA                             ;
                              08    AA    7C 00012    CLRQ    8(BASE)                                         ; 1480
                  03  AA              01    8A 00015    BICB2   #1, 3(BASE)                                     ; 1482
```

```
                                52      04   AC  D0 00019                MOVL    WINDOW, WINDOW_SEGMENT                          ; 1484
                                54      20   A2  D0 0001D 1$:            MOVL    32(WINDOW_SEGMENT), NEXT_SEGMENT                ; 1487
                                            52  DD 00021                PUSHL   WINDOW_SEGMENT                                  ; 1488
                          0000G CF          01  FB 00023                CALLS   #1, DEALLOCATE                                  ;
                                52          54  D0 00028                MOVL    NEXT_SEGMENT, WINDOW_SEGMENT                    ; 1489
                                            F0  12 0002B                BNEQ    1$                                              ; 1491
                          24    A6    AA      02  E0 0002D                BBS     #2, -90(BASE), 3$                               ; 1499
                                      18    A3  B5 00032                TSTW    24(FCB)                                         ; 1506
                                            17  12 00035                BNEQ    2$                                              ;
                                50          53  D0 00037                MOVL    FCB, R0                                         ; 1508
                                        0000G   30 0003A                BSBW    SET_DIRINDX                                     ;
                                0E          50  E8 0003D                BLBS    R0, 2$                                          ;
                                            53  DD 00040                PUSHL   FCB                                             ; 1511
                          0000G CF          01  FB 00042                CALLS   #1, DEL_EXTFCB                                  ;
                                            53  DD 00047                PUSHL   FCB                                             ; 1512
                          0000G CF          01  FB 00049                CALLS   #1, NUKE_HEAD_FCB                               ;
                                      18    AA  DD 0004E 2$:            PUSHL   24(BASE)                                        ; 1515
                          0000G CF          01  FB 00051                CALLS   #1, RELEASE_SERIAL_LOCK                         ;
                                      18    AA  D4 00056 3$:            CLRL    24(BASE)                                        ; 1516
                                            04  00059                 RET                                                     ; 1518

; Routine Size:  90 bytes,    Routine Base:  $CODE$ + 0187

;  533          1519  1
;  534          1520  1 END
;  535          1521  0 ELUDOM
```

```
;                                   PSECT SUMMARY
;
;       Name                    Bytes                           Attributes
;  $CODE$                          481  NOVEC,NOWRT,  RD ,  EXE,NOSHR,  LCL,  REL,  CON,NOPIC,ALIGN(2)


;                                 Library Statistics
;                                          --------  Symbols  --------      Pages       Processing
;       File                               Total     Loaded   Percent       Mapped      Time
;  _$255$DUA28:[SYSLIB]LIB.L32;1           18619         30         0         1000        00:01.9


;                                   COMMAND QUALIFIERS
```

```
;       BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS$:FILUTL/OBJ=OBJ$:FILUTL MSRC$:FILUTL/UPDATE=(ENH$:FILUTL)

; Size:          481 code + 0 data bytes
; Run Time:          00:35.2
; Elapsed Time:      01:31.0
; Lines/CPU Min:      2589
; Lexemes/CPU-Min: 60078
; Memory Used:  229 pages
; Compilation Complete
```

EXTIDX.LIS

GTLCAT.LIS

GETLOC.LIS

EXTEND.LIS

EXTHDR.LIS

FILUTL.LIS

GETREQ.LIS

EXTCONTIG.LIS

ERASE.LIS

GETTIM.LIS

FIND.LIS

GETFIB.LIS

INIFC2.LIS

INIFCP.LIS

EXTFCB.LIS

FILESERV.LIS

FILESIZE.LIS

GETPTR.LIS